Utilisation d'AJAX

AJAX (Asynchronous JavaScript and XML) est une « technologie » proposée par les navigateurs qui permet d'envoyer des requêtes HTTP, celles-ci peuvent être synchrones ou asynchrones (sans avoir à recharger toute la page).

Si vous cliquez sur un lien pour aller sur une page web, vous avez demandé à votre navigateur de faire une requête HTTP de type GET. Cette requête est synchrone, c'est-à-dire que l'on attend d'avoir une réponse complète du serveur.

Dans la majorité des cas, on souhaite utiliser AJAX afin d'apporter du confort à l'utilisateur final, en évitant les rechargements complets de page alors que seule une partie doit être modifiée. Donc AJAX nous permettra de faire des requêtes HTTP asynchrones. C'est-à-dire qu'elles ne vont pas bloquer le reste de la page et seront invisibles aux yeux de l'internaute. Ceci est faisable grâce à l'objet Javascript appelé XMLHttpRequest. Son utilisation est un peu compliquée et dépend du navigateur utilisé, c'est pourquoi la librairie jQuery l'a rendu plus accessible et facile en l'intégrant.

1- Appel serveur en Ajax en mode GET

1.1 - La méthode $.get() :

$.get() charge les données à partir du serveur en utilisant une requête http GET.

Syntaxe : `$.get(URL,data,function(data,status,xhr),dataType)`

Paramètre	Description
URL	Obligatoire. Spécifie l' URL où envoyer la requête.
data	Optionel. Spécifie les données à envoyer au serveur.
function(data,status,xhr)	Optionel. Spécifie une fonction à exécuter si la requête aboutit. Paramètres additionels : • *data* - contient les données résultant de la requête. • *status* - contient le statut de la requête ("success", "notmodified", "error", "timeout", "parsererror") *xhr* - contient l'objet XMLHttpRequest
dataType	Optionel. Spécifie le type de données de la réponse du serveur. Par défaut, jQuery le devine automatiquement. Les types possibles sont : – "xml" - Un document XML – "html" - Du text HTML – "text" - Du text (caractères) – "script" - exécute la réponse en JavaScript, et la retourne en text. – "json" - exécute la réponse JSON, et retourne un objet JavaScript. – "jsonp" - charge un bloc JSON en utilisant JSONP. Rajoute "?callback=?" à l'URL pour spécifier le retour.

Exemples :

- `$.get("test.php");`
- `$.get("test.php", { nom:"Donia", pays:"Tunisie" });`
- `$.get("test.php", { 'couleurs[]' : ["Rouge","Vert","Bleu"] });`
- `$.get("test.php", function(data){ alert("Données: " + data); });`

1.2 - Utilisation de $.get() :

Créez un dossier "tpAjax1" sous le dossier « www » du serveur Web dans lequel copiez le fichier "jquery.js" et créez un fichier "testGet.html" à modifier comme suit:

```
<html>
<head>
<script src="jquery.js"></script>
<script>
$(document).ready(function(){
    $("button").click(function(){
        $.get("test.php", function(data, status){
            alert("Donnees: " + data + "\nStatut: " + status);
        });
    });
});
</script>
</head>
<body>
<button>Envoyer une requête HTTP GET</button>
</body>
</html>
```

Créez le fichier "test.php" dans le dossier "tpEx2" et modifiez le comme suit :

```
<? echo "Ceci est une reponse du serveur
"; ?>
```

Affichez la page "testGet.html" et cliquez sur le bouton.

2 - Appel serveur en Ajax en mode POST

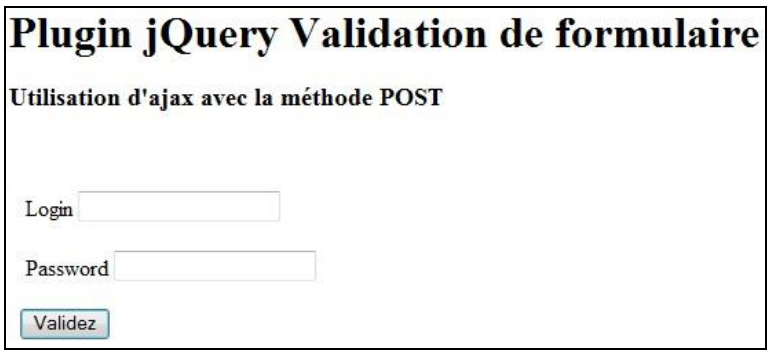
2.1 - La méthode post() :

post() charge les données à partir du serveur en utilisant une requête http POST.

Syntaxe : `$(selecteur).post(URL,data,function(data,status,xhr),dataType)`

2.2 - Utilisation de $.post() :

Créez un nouveau dossier "tpAjax2", dans lequel, copiez les 3 fichiers "jquery.js", "jquery.validate.js" et créer le formulaire "testValid.html" :



A la fin du body, ajoutez une balise div vide et identifiée par "resultat".

Créez le fichier "connect.php" et le modifier comme suit :

```
<?php
    $username = "sonia"; // mettez votre prénom
$password = "guerbouj";  // mettez votre nom     if(
isset($_POST['login']) && isset($_POST['mdp']) ){
        if($_POST['login'] == $username && $_POST['mdp'] == $password){
session_start();
            $_SESSION['user'] = $username;
echo "Succes";
        }
else{
            echo "Echec";
        }
    }
?>
```

Retournez au fichier "TestValid.html" et ajoutez dans l'enveloppe d'exécution après la validation du formulaire le bout de code suivant :

```
$('____________').click(function(){  //selectionnez le bouton
            $.post("_________________",   // URL : nom du fichier php
        {
          login: _____________________, // récupérez le login du form
mdp: _____________________  // récupérez le mdp du from
        },
        function(data,status){
if(data == 'Succes')
     $("#resultat").html("<p>Vous êtes connecté avec "   /p>");
+ status + "<                  else
     $("#resultat").html("<p>"+data+" lors de la connexion! </p>");
},
            'text');  // type de réponse
            });
```

A partir du Serveur Web local, affichez la page testValid.html et testez le formulaire de connexion.